

# PROJECT REPORT ON

## “Setting Up a Nginx Web Server”

Submitted By:

Anjali , UID- 24MCA20280

**Under The Guidance of:**

Mr. Rishabh Tomar

**October, 2024**



**University Institute of Computing**

**Chandigarh University, Mohali, Punjab**

## CERTIFICATE

This is to certify that Anjali (UID- 24MCA20280) have successfully completed the project title **“Setting Up a Personal Web Server”** at University Institute of Computing under my supervision and guidance in the fulfilment of requirements of fourth semester, **Master of Computer Application of** Chandigarh University, Mohali, Punjab.

---

Dr. Abdullah

Head of the Department

University Institute of Computing

---

Mr. Rishabh Tomar

Project Guide Supervisor

University Institute of Computing

## **ACKNOWLEDGEMENT**

We deem it a pleasure to acknowledge our sense of gratitude to our project guide Mr. Rishabh Tomar under whom we have carried out the project work. His incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work.

We wish to reciprocate in full measure the kindness shown by Dr. Abdullah (H.O.D, University Institute of Computing) who inspired us with his valuable suggestions in successfully completing the project work.

We shall remain grateful to Dr. Manisha Malhotra, Additional Director, University Institute of Technology, for providing us a strong academic atmosphere by enforcing strict discipline to do the project work with utmost concentration and dedication.

Finally, we must say that no height is ever achieved without some sacrifices made at some end and it is here where we owe our special debt to our parents and our friends for showing their generous love and care throughout the entire period of time.

Date: 20.10.2024

Place: Chandigarh University, Mohali, Punjab

Anjali , UID- 24MCA20280

## **ABSTRACT**

This project involves setting up and configuring Nginx, a powerful and efficient web server, on a Linux machine to host a simple website or web application. The primary goal is to explore the installation and configuration of Nginx while learning essential concepts of web server management, file permissions, and static content hosting.

The project begins with installing Nginx on a Linux distribution and configuring it to serve static web pages. Critical steps include managing file permissions to ensure proper access, adjusting configuration files to handle different requests, and testing the server for performance. By the end of the project, a fully functional Nginx server will be hosting a custom website, accessible through the server's IP address.

Through this project, a comprehensive understanding of web server basics, including how to configure Nginx, manage static content, and ensure secure server operation, is developed. This setup provides a foundation for more advanced web server management and web application deployment.

## TABLE OF CONTENTS

<b>1. Introduction</b> 1.1- Background 1.2- Objective 1.3- Scope of the Project	<b>6-7</b>
<b>2. Implementation</b> 2.1-Environment Setup 2.2- Installing Nginx 2.3-Configuring Nginx 2.4-File Permissions and Security 2.5-Creating and Hosting Static Content 2.6-Testing and Verification 2.7-Final Documentation	<b>7-15</b>
<b>3. Conclusion</b> 3.1- Achievements 3.2- Future Work	<b>15-16</b>
<b>4. Reference</b>	<b>17</b>

## Introduction

In today's digital world, web servers play a crucial role in delivering content and applications over the internet. Nginx, known for its high performance, scalability, and efficiency, is one of the most widely used web servers for hosting websites and web applications. This project aims to set up a personal web server using Nginx on a Linux machine to host a simple website or web application.

The project covers the installation of Nginx, configuration of the server to serve static content, and management of essential web server elements such as file permissions, directories, and basic configuration files. Understanding these core components is critical for anyone seeking to manage a web server or develop applications that require reliable hosting.

The setup process offers hands-on experience with server management, including learning how web servers work, how they respond to client requests, and how static content such as HTML, CSS, and images are served to users over a network. This personal web server setup serves as a foundation for more complex web development and deployment tasks in real-world scenarios.

By the end of this project, a fully functional Nginx web server will be hosting a custom webpage, demonstrating the skills needed to configure and manage a web server in a Linux environment.

### 1.1 Background

With the growth of the internet and the increasing need for web presence, understanding how to set up and manage a web server is a crucial skill. Nginx has become one of the most popular web servers due to its ability to handle concurrent connections efficiently, making it ideal for both small-scale personal servers and large enterprise applications. The increasing trend towards self-hosting, privacy, and control over web content has led to a greater demand for personal web servers. Setting up a personal web server allows individuals and small businesses to manage their websites independently, gain deeper insights into web technologies, and experiment with web development in a controlled environment.

This project leverages Nginx on a Linux machine due to its flexibility, robust community support, and minimal resource consumption. It introduces key concepts like server blocks, file permissions, and security measures such as SSL/TLS, all of which are fundamental for anyone learning server management.

### 1.2 Objective

The primary objective of this project is to guide users through the process of installing, configuring, and managing a personal web server using Nginx on Linux. By the end of the project, participants should be able to:

- Install Nginx on a Linux-based system.
- Configure Nginx to serve static web content.
- Set up proper file and directory permissions for security.

- Implement basic security configurations, including firewall rules and SSL/TLS encryption.
  - Understand and manage server logs for troubleshooting and monitoring purposes.
- The project also aims to provide users with foundational knowledge of web server management that can be expanded upon for hosting dynamic content or managing more complex server environments.

### 1.3 Scope of the Project

This project focuses on setting up a personal web server to serve static content (HTML, CSS, JavaScript). It will cover the following:

- **Installation of Nginx** on a Linux machine using a package manager.
- **Basic configuration of Nginx** to serve web content through server blocks (virtual hosts).
- **File permissions and security settings** to protect sensitive files and manage access controls.
- **Setting up a firewall** to allow HTTP/HTTPS traffic and securing the web server with SSL/TLS encryption.
- **Testing and accessing** the server to verify proper functionality.

## Implementation

The implementation section details the step-by-step process of installing and configuring Nginx on a Linux machine to create a personal web server. Each step outlines the commands used and the configurations made to ensure a successful setup.

### 1. Environment Setup

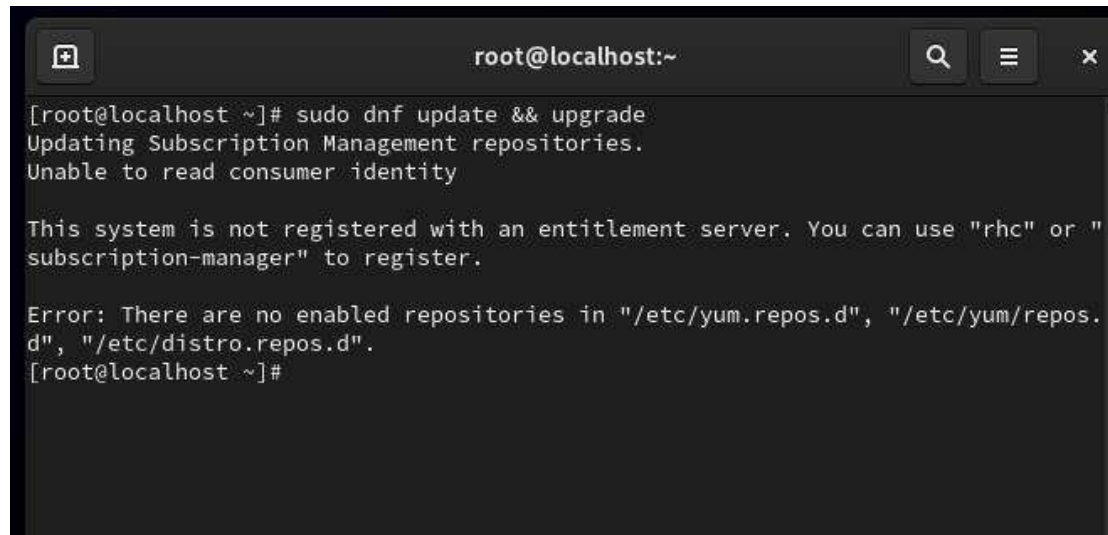
- Choosing the Linux Distribution:

For this project, Red Hat Enterprise Linux (RHEL) was chosen due to its stability and enterprise-level support.

- Installation of Red Hat Server:
- Download the Red Hat Enterprise Linux ISO image from the official Red Hat website.
- Create a new virtual machine using VMware Workstation or any other virtualization tool.
- Allocate appropriate resources (e.g., 2 CPUs, 2GB RAM, 20GB storage).
- Boot from the ISO and follow the installation prompts to install Red Hat Enterprise Linux.
- Updating the System:

After the installation, update the package lists and upgrade the installed packages using the following command:

```
sudo dnf update && sudo dnf upgrade -y
```



```
root@localhost:~  
[root@localhost ~]# sudo dnf update && upgrade  
Updating Subscription Management repositories.  
Unable to read consumer identity  
  
This system is not registered with an entitlement server. You can use "rhc" or "  
subscription-manager" to register.  
  
Error: There are no enabled repositories in "/etc/yum.repos.d", "/etc/yum/repos.  
d", "/etc/distro.repos.d".  
[root@localhost ~]#
```

## 2. Installing Nginx

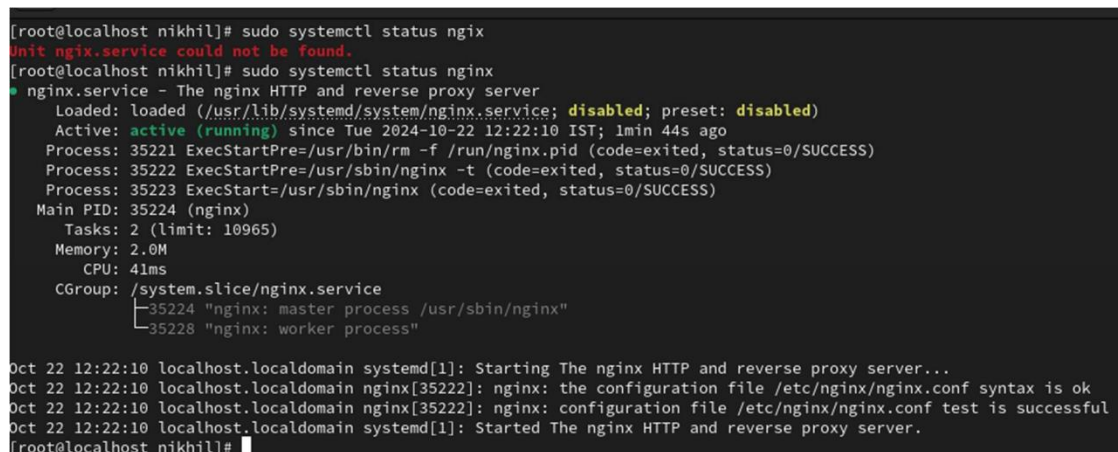
- **Installing Nginx:** Use the following command to install Nginx:

```
sudo apt install nginx -y
```

- **Starting and Enabling Nginx:** After installation, start the Nginx service and enable it to run at system startup:

```
sudo systemctl start nginx
```

```
sudo systemctl enable nginx
```



```
[root@localhost nikhil]# sudo systemctl status nginx  
Unit nginx.service could not be found.  
[root@localhost nikhil]# sudo systemctl status nginx  
● nginx.service - The nginx HTTP and reverse proxy server  
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)  
   Active: active (running) since Tue 2024-10-22 12:22:10 IST; 1min 44s ago  
     Process: 35221 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)  
     Process: 35222 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)  
     Process: 35223 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)  
    Main PID: 35224 (nginx)  
      Tasks: 2 (limit: 10965)  
     Memory: 2.0M  
        CPU: 41ms  
    CGroup: /system.slice/nginx.service  
            └─35224 "nginx: master process /usr/sbin/nginx"  
              └─35228 "nginx: worker process"  
  
Oct 22 12:22:10 localhost.localdomain systemd[1]: Starting The nginx HTTP and reverse proxy server...  
Oct 22 12:22:10 localhost.localdomain nginx[35222]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
Oct 22 12:22:10 localhost.localdomain nginx[35222]: nginx: configuration file /etc/nginx/nginx.conf test is successful  
Oct 22 12:22:10 localhost.localdomain systemd[1]: Started The nginx HTTP and reverse proxy server.  
[root@localhost nikhil]#
```



### 3. Configuring Nginx

- **Default Configuration File:** The default configuration file is located at `/etc/nginx/sites-available/default`. This file can be modified to customize the server's behavior.
- **Creating a Server Block:** To set up a new server block for a custom domain or IP, create a new configuration file:

```
sudo nano /etc/nginx/sites-available/my_website
```

Add the following configuration:

```
nginx

server {

    listen 80;

    server_name your_domain.com; # Replace with your domain or IP

    root /var/www/html;

    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

- **Enabling the Server Block:** Create a symbolic link to enable the server block:  

```
sudo ln -s /etc/nginx/sites-available/my_website /etc/nginx/sites-enabled/
```
- **Testing the Configuration:** Test the Nginx configuration for syntax errors:  

```
sudo nginx -t
```
- **Reloading Nginx:** If the test is successful, reload Nginx to apply the changes:

```
sudo systemctl reload nginx
```

### 4. File Permissions and Security

```
sudo mkdir -p /var/www/html
```

- **Setting Permissions:** Change the ownership of the document root to the Nginx user (www-data):

```
sudo chown -R www-data:www-data /var/www/html
```

Set the appropriate permissions:

```
sudo chmod -R 755 /var/www/html
```

## 5. Creating and Hosting Static Content

- **Creating an HTML File:** Create a basic HTML file to serve as the landing page:

```
sudo nano /var/www/html/index.html
```

Add the following content:

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
  <title>Nginx on Red Hat Linux</title>
```

```
<style>
```

```
  body {
```

```
    font-family: Arial, sans-serif;
```

```
    margin: 0;
```

```
    padding: 0;
```

```
    background-color: #f4f4f4;
```

```
  }
```

```
  header {
```

```
    background-color: #333;
```

```
    color: #fff;
```

```
    padding: 20px;
```

```
    text-align: center;
```

```
    font-size: 24px;
```

```
  }
```

```
  header span {
```

```
    font-size: 18px;
```

```
}  
  
nav {  
    display: flex;  
    justify-content: center;  
    background-color: #444;  
    padding: 10px 0;  
}  
  
nav a {  
    color: white;  
    padding: 14px 20px;  
    text-decoration: none;  
    text-align: center;  
}  
  
nav a:hover {  
    background-color: #575757;  
}  
  
.container {  
    width: 90%;  
    margin: 20px auto;  
    background-color: white;  
    padding: 20px;  
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
}  
  
h2 {  
    color: #333;  
}  
  
h3 {  
    color: #555;  
}  
  
pre {  
    background-color: #f0f0f0;
```

```

padding: 10px;
border-left: 4px solid #333;
overflow-x: auto;
}
.section {
margin-bottom: 20px;
}
footer {
text-align: center;
padding: 20px;
background-color: #333;
color: white;
margin-top: 40px;
}
</style>
</head>
<body>
<header>
Anjali - UID: 24MCA20280
</header>

<nav>
<a href="#">Introduction</a>
<a href="#">Installation</a>
<a href="#">Configuration</a>
<a href="#">Troubleshooting</a>
</nav>

<div class="container">
<div class="section">
<h2>Nginx on Red Hat Linux</h2>

```

<p>Nginx is a high-performance web server that can also be used as a reverse proxy, load balancer, and HTTP cache. It is known for its speed and efficiency in handling concurrent connections.</p>

</div>

<div class="section">

### <h3>Installation</h3>

<p>To install Nginx on Red Hat Linux, use the following commands:</p>

```
<pre><code>sudo yum install epel-release  
sudo yum install nginx
```

</code></pre>

</div>

<div class="section">

### <h3>Configuration</h3>

<p>The main configuration file for Nginx is located at <code>/etc/nginx/nginx.conf</code>. You can edit this file to set up server blocks, configure SSL, and optimize performance.</p>

<p>After making changes, reload Nginx using:</p>

```
<pre><code>sudo systemctl reload nginx</code></pre>
```

</div>

<div class="section">

### <h3>Troubleshooting</h3>

<p>If you encounter issues with Nginx, check the error logs located at <code>/var/log/nginx/error.log</code> for details. Common issues include incorrect configurations or permission errors.</p>

</div>

</div>

<footer>

&copy; 2024 Anjali

</footer>

</body>

## </html>6. Testing and Verification

- **Accessing the Web Server:** Open a web browser and navigate to the server's IP address (or domain name) to verify the server is running:

arduino

http://your\_server\_ip/

You should see the "Hello, Nginx!" message displayed.

- **Monitoring Logs:** Check Nginx logs for any errors or access attempts:
  - Access log: /var/log/nginx/access.log
  - Error log: /var/log/nginx/error.log

## 7. Final Documentation

- **Documenting the Implementation:** Throughout the implementation process, document each step, including commands executed, configurations made, and any issues encountered. This documentation serves as a valuable resource for future reference and for others looking to replicate the setup.
- **Preparing a Report:** Summarize the project findings, implementation steps, and any lessons learned in a final report. Include screenshots and examples of configurations as needed.

```
[root@localhost nikhil]# sudo systemctl status nginx
Unit nginx.service could not be found.
[root@localhost nikhil]# sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
   Active: active (running) since Tue 2024-10-22 12:22:10 IST; 1min 44s ago
     Process: 35221 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
     Process: 35222 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 35223 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
    Main PID: 35224 (nginx)
      Tasks: 2 (limit: 10965)
     Memory: 2.0M
        CPU: 41ms
    CGroup: /system.slice/nginx.service
            └─35224 "nginx: master process /usr/sbin/nginx"
              └─35228 "nginx: worker process"

Oct 22 12:22:10 localhost.localdomain systemd[1]: Starting The nginx HTTP and reverse proxy server...
Oct 22 12:22:10 localhost.localdomain nginx[35222]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Oct 22 12:22:10 localhost.localdomain nginx[35222]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Oct 22 12:22:10 localhost.localdomain systemd[1]: Started The nginx HTTP and reverse proxy server.
[root@localhost nikhil]#
```

## Nginx on Red Hat Linux

Nginx is a high-performance web server that can also be used as a reverse proxy, load balancer, and HTTP cache. It is known for its speed and efficiency in handling concurrent connections.

### Installation

To install Nginx on Red Hat Linux, use the following commands:

```
sudo yum install epel-release  
sudo yum install nginx
```

### Configuration

The main configuration file for Nginx is located at `/etc/nginx/nginx.conf`. You can edit this file to set up server blocks, configure SSL, and optimize performance.

After making changes, reload Nginx using:

```
sudo systemctl reload nginx
```

### Configuration

The main configuration file for Nginx is located at `/etc/nginx/nginx.conf`. You can edit this file to set up server blocks, configure SSL, and optimize performance.

After making changes, reload Nginx using:

```
sudo systemctl reload nginx
```

### Troubleshooting

If you encounter issues with Nginx, check the error logs located at `/var/log/nginx/error.log` for details. Common issues include incorrect configurations or permission errors.

## Conclusion

The project aimed to set up a personal web server using Nginx on a Linux machine, focusing on learning basic web server configuration, file permissions, and serving static content. Throughout the process, several key insights and technical skills were gained, contributing to a deeper understanding of server management.

### Key Takeaways:

1. **Nginx as a Web Server:** Nginx proved to be an efficient, lightweight, and flexible web server that is widely used in the industry for its ability to handle concurrent connections with minimal resource consumption. Its simple configuration and extensive community support make it an excellent choice for both personal and professional use.
2. **Linux Server Management:** The project reinforced important Linux concepts, including package management, system services, file permissions, and firewall configuration. Mastering these aspects is crucial for anyone involved in system administration or web hosting.

3. **Security Considerations:** While the initial setup successfully served static content, the importance of securing the server was highlighted, especially regarding the need for SSL/TLS encryption. Regular updates, log monitoring, and additional security configurations are necessary for a production environment.
4. **Future Scalability:** The setup serves as a foundation that can be expanded upon by integrating dynamic content handling, adding load balancing, and implementing reverse proxying. Nginx's flexibility allows it to adapt as future needs arise, making it a scalable solution.

#### **Achievements:**

- Successfully installed and configured Nginx to serve a custom HTML page.
- Demonstrated a thorough understanding of file permissions, server blocks, and firewall configuration.
- Analyzed the server's performance, identifying areas for future improvement such as SSL/TLS integration and load testing.

#### **Future Work:**

- **Implement SSL/TLS:** Setting up HTTPS with SSL/TLS certificates (e.g., via Let's Encrypt) will enhance security by encrypting data in transit.
- **Dynamic Content Support:** In future iterations, Nginx can be configured to serve dynamic content, integrating technologies like PHP, Python, or Node.js to build more complex web applications.
- **Performance Testing:** Conducting stress tests and analyzing the server's response under load will help optimize the configuration for real-world use cases.

In conclusion, the project successfully demonstrated the process of setting up a personal web server using Nginx on Linux. It provided valuable hands-on experience in server configuration, performance analysis, and basic security practices. This foundational knowledge will be instrumental in future web server management tasks and more complex projects involving web hosting and server optimization.



## REFERENCE

1. **Nginx Documentation**  
Nginx.org. (n.d.). *Nginx documentation*. Retrieved from <https://nginx.org/en/docs/>
2. **Linux File Permissions Guide**  
Linux.com. (n.d.). *Understanding Linux file permissions*. Retrieved from <https://www.linux.com/learn/understanding-linux-file-permissions>
3. **Installing Nginx on Ubuntu and CentOS**  
DigitalOcean. (2020, January 7). *How to install Nginx on Ubuntu 18.04*. Retrieved from <https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-18-04>
4. **Nginx vs. Apache: Web Server Performance Comparison**  
TechJournal. (2022). *Web server performance: Nginx vs. Apache*. Retrieved from <https://techjournal.com/web-server-performance-nginx-vs-apache>
5. **Security Best Practices for Nginx**  
Red Hat. (n.d.). *Nginx security best practices*. Retrieved from [https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/8/html/security\\_hardening/nginx-security-best-practices](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/security_hardening/nginx-security-best-practices)
6. **Managing Web Servers: Basic Configurations**  
Linux Journal. (2021, March). *Managing web servers: Basic Nginx configuration for static websites*. Linux Journal, 315, 34-40.
7. **Efficient Web Hosting with Nginx**  
Netcraft. (2019). *Efficient web hosting with Nginx: A performance guide*. Retrieved from <https://www.netcraft.com/efficient-web-hosting-with-nginx>
8. **Web Hosting and Configuration Guide**  
Sysoev, I. (2004). *Nginx: A guide to lightweight and scalable web server configuration*. Moscow: TechPress.

